5-1976

Computer assisted time series analysis (CATSA)

William A. Dunn
Pacific University

Computer assisted time series analysis (CATSA)

Abstract

Computer assisted time series analysis (CATSA)

Degree Type

Thesis

Degree Name

Master of Science in Vision Science

Committee Chair

Clifton M. Schor

Subject Categories

Optometry

Computer Assisted Time Series Analysis (CATSA)

CATSA

This is submitted
in partial fulfillment
of the requirements for the degree
Doctor of Optometry

Submitted to:

Pacific University College of Optometry

Submitted by:

William A. Dunn
May, 1976

This Thesis for the Doctor of Optometry Degree by

William A. Dunn

has been approved by

Clifton M. Schor

May 5, 1976

TABLE OF CONTENTS

Forward

This monograph describes a computer system which was designed for the on line analysis of research data. As scientific concepts have become more sophisticated so has the instrumentation necessary to measure these new quantities. As a particular example, frequency domain, a once esoteric consideration is now a commonly used phrase. This has been made possible largely through the development of inexpensive, fast digital computers and a parallel development in the mathematics of time series analysis, notably, the fast Fourier Transform. This computer system (CATSA) allows the advanced researcher to explore his area of concern with the aid of these new advancements in instrumentation.

This document was written for the reader who has had some experience with computers and a conceptual understanding of the Fourier Transform. No assumption of any computer language was assumed, however, since this system uses a unique language scheme that is developed in detail within the text.

Computer Assisted Time Series Analysis (CATSA)

1. Introduction

1.1 The Research Computer

Computers are similar to modern calculators in that they accurately perform arithmetic operations. Unlike most calculators though computers can automatically sequence through a number of operations without operator intervention. The use of automated calculators is obvious in areas such as banking and statistics where numbers are in evidence and arithmetic is in apparent use. Computer utility may not be as evident in the scientific laboratory since often only subtle continuously varying physical changes are occurring. These physical changes can be converted to numerical variations by instruments called analog to digital converters and, therefore, allow computers to monitor the progress of an experiment.

Of frequent interest to the researcher is how time varying quantities are affected under different conditions. Time varying quantities are easily studied by computers since data can be gathered and studied at great length using powerful mathematical tools.

1.2 Thesis Objective

The objective of this thesis is to provide a computer system that will allow researchers the advantage of computer assistance with a minimum of preparation time.

To meet this broad goal the system has to be quickly adaptable to many situations by providing "on-the-spot" programming capability. The instructions of the program have to be individually powerful yet no so broad that uniqueness of function is lost. The ability to provide various forms of hard copy is necessary for further analysis and the ability to store previously written programs is essential for a multiuser system.

1.3 Overview of CATSA

The program is called CATSA (Computer Assisted Time Series Analysis) and was designed around a Nova 800 computer with analog to digital (A/D) and digital to analog (D/A) peripheral equipment. This specific system uses a 12 Bit A/D converter with eight channels of multiplexed inputs for acquiring the analog signals and a two channel 10 Bit D/A converter which is used to drive a storage oscilloscope display. Other peripheral devices included a teletype, high speed paper tape punch, and a paper tape reader. The interface equipment for the A/D and D/A converters were in-house designs of Pacific University and allow data acquisition to be synchronized at the hardware level by external clocks, for greater accuracy.

CATSA is a block manipulation program and performs commands either individually from the teletype or from stored instruction strings called macro commands. All commands perform manipulations on an entire collection of data at one time (a block) so that the user is able to "think" in terms of time periods (epoch) instead of individual "bits" of information. The program is fully operator interactive through commands and displays and quickly adaptable (through macro manipulations) to change as the needs arise. Commands range from simple copying of data from one block to another through complex Fourier transformation. Assembly language subroutines can be easily added to this system so that an expansion to fit a particular need is inexpensively implemented

2. Time Series Analysis Primer

2.1 Introduction to Digitizing

Time varying quantities can range from the smooth pendulum motions through the erratic movement of a heated molecule. In order to analyze any of these on a mathematical basis they have to be reduced to either a formula

or some other rational description. In the case of the molecule's position a sequence of numbers would have to suffice as the description since it is a random process.

A digital computer is inherently a discontinuous analyzer of data since the functioning of the machine is divided into bits (binary or base 2 digits) and these into groups called words. A means is thus necessary to break the real world (analog) quantities (such as movement, position, or volts) into discontinuous (digital) groups for computer analysis. One way is to simply record the magnitude of the quantity (sample) at equal intervals and have these values entered into the computer. This sampling process is referred to as digitizing.

Intuitively, if we sample frequently enough the digitizing process will represent the analog data so faithfully that no ambiguity will exist (that is that no other analog wave form could be confused with the digital representation). The question then arises of how fast do we have to sample to meet this criterion of non-ambiguity? This question is best answered by first looking at other ways to describe a time varying quantity.

2.2 The Frequency Domain

Any wave form (the time varying quantity) can be constructed from a simple summation of other wave forms. Fourier developed a means of expressing any wave form as a sum of sinusoids of various amplitudes and phases. When a wave is expressed in this way it is said to be represented in the frequency domain. Any wave form, thus, has "frequency components" and the highest frequency component or the highest Fourier sine wave that is necessary to characterize a given wave form is the clue to the highest sampling rate that we must maintain. Since a sine wave is assumed to be the simple wave form it,

therefore, needs only two samples to characterize each cycle (this is obvious since these two samples will completely describe its amplitude, frequency, and phase). We, therefore, need a minimum theoretical sampling rate of 2 times the highest frequency of the wave form being digitized. This is referred to as the Nyquest, folding, or aliasing rate (Reference 3 page 31). Although the Nyquest rate is the minimum sampling frequency, various sources of error require a practical minimum sampling frequency of 2.5 times (Reference 3 Page 32) the highest wave form frequency.

3. Description of CATSA

3.1 Definitions:

Block - All data is manipulated as a large group called a block. The size of this group is variable and is specified by the user. Each block is further divided into two equal parts and arbitrarily named the real and imaginary halves. Mathematically, a complex number consists of a pair of parts, one is called real and the other imaginary. When the block operation of addition is performed, for instance, the first word of Block 1 is added to the first word of Block 3 and the results stored in the first location of Block 3. The second word of Block 1 is then added to the second word of Block 3 and the results stored back into the second location of Block 3, and so on.

Real and Imaginary - A vector quantity can be expressed by two values, its phase and its magnitude. This vector can also be expressed as a complex number consisting of a real and an imaginary component. In this document the terms real and imaginary refer to these components of a complex number.

Parameter - A constant which is used by a command routine and is set to value at some previous time.

Word - A computer word in this system is 16 binary bits and expresses the value of a single number.

Command - A CATSA command consists of an alphabetic letter and depending on the function up to two following numbers. A command directs the computer to perform some single instruction such as adding one block to another block. The commands may be executed individually or grouped together for programmed, sequential execution as a "macro" string.

Bin Number - A bin is a computer word in a block. The bin number is simply a count of how far from the beginning this bin is in a block. The first bin is bin number 1, the second is bin number 2, and so on.

3.2 General Description

There are basically five classes of instructions for block manipulation: 1) arithmetic, 2) acquisition, 3) frequency domain, 4) display, and 5) housekeeping. Commands which implement instructions consist of an alphabetic letter or (symbol) and may be followed by one or two numbers. Since the numbers refine the instruction in some way they are referred to as attributes.

3.3 Block Manipulation Instructions

3.3.1 C(n) -(COPY)- This instruction requires one attribute, a 2 or 3 and causes data from block one to be copied into block 2 or 3 as specified in the attribute. The previous contents of block 1 are unchanged.

3.3.2 S(n) (Swap) - This instruction requires one attribute, a 2 or 3 and implements an exchange between blocks 1 and n (The attribute). The total contents of block one are transferred to n and the total contents of n are transferred to one.

3.3.3 H(n) (half) - This program requires one attribute, a 1, 2, or 3. Execution causes the real and imaginary parts of the specified block to be exchanged with one another.

3.3.4 E (Expunge) - This instruction clears all blocks to zero.

3.3.5 J (Expunge imaginary) - The imaginary portion of block 1 is cleared to zero.

3.4 Arithmetic Instructions

3.4.1 + (Add)- This instruction adds block 1 to block 3 and places the result into block 3, block 1 remains unaltered.

3.4.2 - (Sub)- This command causes block 1 to be subtracted from block 3 and the results to be placed into block 3, block 1 remains unaltered

3.4.3 x (mult) - This instruction causes block 1 to be multiplied by block 2 with the results placed into block 1, block 2 remains unaltered.

3.4.4 NO(n) (Normalize) - This command uses two attributes. The first is always zero and the second sets the normalizing value. The legal range for n is 1 through 9. Normalizing is accomplished by dividing every word in block 1 by 2^n.

3.4.5 K(integral) - This command has no attributes. Execution replaces block 1 (real with the integral of block 1 (real). For example, a constant value (no zero) in block 1 prior to K would result in a linearly increase function (beginning to end).

3.5 Data Aquisition Commands

3.5.1 G(n) (Get) - Up to 3 channels of information can be acquired simultaneously (within several microseconds of one another, for corresponding bin numbers). Channel one is placed in block 1, Channel 2 in block 3, and Channel 3 in block 3. Only the real portion of the block is affected and the previous contents are lost. The attributes are 1, 2, or 3. If n=1, one channel of data is acquired from Channel 1 to block 1. If n=2 two blocks are acquired, Channels 1 and 2. For n=3, three blocks are acquired from Channels 1, 2, or 3.

After execution of the command data is stored into the appropriate location at the positive transition of an external oscillator. The oscillator is connected directly to the interface so accurate timing is maintained. This oscillator is referred to as the digitizing clock.

The extreme input analog voltage range is 10V (4095_{10} is the digital representation).

3.5.2 T (trigger) - This command when excited monitors multiplexor Channel 4 for a positive transition. As soon as a transition is observed the next instruction in the macro string is executed. As long as no changes are occurring on line 4 a steady glow can be observed on Multiplex (MUX) address 4. This can serve as a cue to the operator that the computer is awaiting a triggering event.

3.6 Frequency Domain Commands

3.6.1 F (FFT) - This command implements a fast Fourier Transform on block 1 data. The real portion is assumed to be positive time starting with time zero in bin number 1 and increasing time with increasing bin numbers. The imaginary part is assumed to be negative time with the largest bin number representing the time farthest from zero. With normal operation only the real block is filled with data while the imaginary part should be filled with zeros.

At the completion of the FFT, bin number 1 represents the DC component, bin number 2 represents the first Fourier coefficient and so on. Remember that the FFT will have represented unambiguously only 1/2 as many coefficients as time samples. This is important to bear in mind when computing the actual frequency represented by a particular coefficient.

Example: Assume bin size = 16

Clock rate 25 Hz

This means that 16 samples were acquired with an interval between each one of 1/25 Hz=.04 sec. The highest frequency that could have been represented properly was 25/2 Hz=12.5 Hz. Since only $16/2^{(8)}$ samples would represent coefficients in the FFT, each coefficient would represent 12.5/8 Hz or 1.5625 Hz. That is, the following frequencies would be represented by these numbers:

Bin #	Freq.
1	0 (DC component)
2	1.5625 Hz
3	3.125 Hz
.	.
.	.
.	.
8	10.9375 Hz
9	12.5 Hz

NOTE: $\frac{\text{Clock rate}}{\text{bin size}} = \text{bin resolution Hz}$

In this case the FFT result yields a similar result to the power output of eight filters each with a 1.56 Hz bandwidth, which were excited by the signal represented in block 1.

3.6.2 I (Inverse FFT) - This program performs the inverse FFT on the data in the real and imaginary parts of block 1. This function is useful to transform the data from frequency domain to time domain. This is also useful as part of rapid implementation of the convolution integral for digital filtering (see appendix).

3.6.3 0_0(Smooth) - This command when executed performs a three point low pass filter function. This function tends to reduce the errors caused in the FFT by the abrupt transition at t=0 and t=max. This error is caused by the inherent assumption in the FFT that the data within the block is periodic.

In general real data does not follow this criterion. Only block 1 real is affected by this command and 0 is the required attribute. The algorithm used in this implementation is $t_n = 1/4\ t_{n-1} = 1/2\ t_n - 1/4\ t_{n+1}$. At both ends of the block t_{n-1} or t_{n+1} is replaced by t_n.

3.6.4 A (Absolute value) - This command performs the absolute value function of the real and imaginary parts of block 1 and places the result in block 1 real. This is performed by implementing: $\sqrt{Re^2 + Im^2}$ on a point by point basis. The purpose for doing this is to extract the power vector from the real and imaginary parts. A power spectrum would be properly performed on a block of data by implementing the following command: F, A, 0.

3.6.5 N1:(Threshold crossing period acquisition) - The N1 is used with N2 to perform frequency and amplitude histograms. This algorithm acquires 1 channel of data into block 1 by counting a clock pulse (digitized clock) and recording the amount of time experienced between successive positive transitions on the data channel. The transition point is selected by the operator and entered as parameter P5. The time between transitions is recorded into block 1, imaginary. For greater accuracy, record keeping starts after the second transition. The peak amplitude between zero crossings is recorded in the imaginary part of block 1.

3.6.6 N2: (Amplitude histogram) - Execution of this algorithm takes the data in block 1 real (which was acquired by N1, G1, or other means), divides it by the parameter P6 and accumulates in the corresponding bin number in

block 2, real. If the value is larger than 1 the largest bin number the last bin number is incremented, indicating an overflow condition

Using N1 and N2 the frequency of a simple sinusoid or other single component wave form can be determined easily with high resolution. High frequency components will be close to bin numbers 1 and low frequency components will appear at the opposite end. The following example will demonstrate how this is calibrated:

Example: Assume: Block size = 16

Clock rate = 25 Hz

P6 = 3

NOTE: bin resolution = $\frac{\text{clock rate x P6}}{\text{L-1}}$

This means the first bin will represent 3(P6) cycles of a 25 Hz rate or 75 Hz. The second bin would represent 3(P6)/2 (bin number) x 25 Hz = 37.5 Hz

The maximum frequency = 75 Hz

The minimum frequency = 75/15 - 5 Hz

3.7 Display Commands

3.7.1 Y (n) (amplitude scaling) - This command when implemented searches the data in block 1 to determine the maximum and minimum values, then scales them to a new maximum of 1023 and minimum of zero. One attribute follows the command and must be in the range of 1-4. The meaning of the attribute is as follows:

1 - Scan for maximum and minimum values in the real portion only and then scale only the real portion.

2 - Scan for maximum and minimum values in the total block (both real and imaginary portions) and then scale the total block.

3 - Only scale the real portion using a previously found maximum and minimum value. That is don't scan for a new maximum and minimum.

4 - Scale the total block (both real and imaginary halves) using a previously found maximum and minimum.

3.7.2 V (n)(View) - This command causes data to be displayed on the oscilloscope, pen recorder, or paper tape. The attribute specifies the output device and action to be taken. The attribute is interpreted in the following manner:

0 - Erase the storage display.

1 - Horizontally scale the data so that it will fit into the CRT and plot the data in block 1 real using a conventional rectilinear coordinate scheme. The data is assumed to have been previously scaled to a minimum of 0 and a maximum of 1023. Any values which exceed this range are plotted at the limit resulting in a "flat top" so that the operator can see that an error condition existed. All points are connected by a smooth line so that even when a few points are plotted the trends between points are highly visible.

The determination of the horizontal scale factor is automatic and proceeds with integer precision (no fractional component). This results in the horizontal display portion varying in length according to the block size specification. At worst this will be about 1/2 the full width of the CRT.

2 - This attribute will cause the data in block 1 to be plotted as a "scatter-graph". That is, paired bin numbers in the real and imaginary halves are plotted against one another with real on the vertical axis and imaginary on the horizontal axis (each point is plotted thusly; $real_n$, $Imag_n$, where n = the bin number). Only the first quadrant is displayed. Therefore, a Y2 or Y4 command is assumed to have preceded this command.

Such a display is useful for displaying the histogram results (amplitude vs. frequency components of N1 and N2) and the FFT results. In the FFT case the plot is really a polar plot since each point represents the head of a vector.

3 - This attribute will cause a dashed horizontal reference line to be drawn. This line is positioned vertically at the point where "zero" was found during the scaling operation, Y, and is a zero reference. This line is plotted with small vertical"tags" every tenth bin number as a reference to the operator. If zero was not within the range of the maximum and minimum during the scaling scan (all values are positive or all are negative) no line is plotted.

4 - This attribute is used to display data on a chart recorder. The action is similar to V1 except the rate of display is slowed down and a "device ready" signal precedes the display. The "device ready" signal is merely a rise in the output voltage of .5 volts held constant for .1 sec. This signal is sufficient to trigger the start of the motor in the chart recorder. At the end of this command the voltage is returned to zero which again signals the recorder to turn the motor off after .5 seconds in this zero state.

5 - This attribute is used to output data from block 1 to paper tape. Either the console teletype tape punch, or a high speed paper tape punch can be selected by setting data switch zero on the front of the computer. Switch zero on (up) selects high speed. 64 spaces of leader and trailer are punched before and after the data. Both real and imaginary parts of block 1 are punched. Each word requires two punch positions with the most insignificant portion (Bits 8-15) punched first.
To signify the start of data two sets of the octal word 377 (all ones) are punched to separate the data from the leader. To allow a full reconstruction of the machines' status at the time the data was prepared the following constants are next punched:
BKS (Block Size)
SCL3 (Maximum value found during vert scale)

SCL2 (minimum value found during vert scale)

P2 (the time parameter)

P6 (the bin scale factor)

These constants are followed by 4 sets of nulls then the data, real followed by imaginary. At the completion of this task the carriage will return on the console typewriter and the user may type a comment which will be punched on the paper tape. This comment is limited to 50 characters and is terminated by the user by typing the escape key. In the case of the high speed unit the comment is punched after the escape key is typed, the low speed units comment is punched at the same time as the comment is originated. The escape character is punched at the end of the comment and 64 more characters of nulls are punched as "trailer".

3.7.3 D(n) or DO (n) (Digital Summary) - Execution of this command causes the data in the real portion of block 1 to be searched for inflection points (the point where the slope of the curve changes from positive to negative or negative to positive). To assist the operator in disregarding minor inflections this process can take place by first averaging a number of points together (the attribute value when within the range 1-9).

The display is written on the CRT (a console teletypewriter if the command is in the form DOO) in the following form:

The maximum and minimum values determined in the previous amplitude scaling command followed by the inflection points in four columns. The first column represents the amplitude of the positive to negative slope

inflections. The second column is the bin number associated with this inflection multiplied by the parameter P2. The third column represents the amplitude of the negative to positive slope inflections and the last column represents the bin number associated with this inflection multiplied again by the parameter P2. The purpose in multiplying the bin number by the parameter P2 is to directly relate the inflection with time. This occurs properly when P2 represents the digitizer clocks period (the dimension (sec. or ms.) is chosen to make P2 a whole number).

This program assumed that all values are positive or zero (This is easily satisfied by scaling the data before performing this operation. If an average of 9 points is not sufficient for a given set of data any number can be used by first specifying the averaging interval in parameter P4 and then using DO1).

3.8 Housekeeping Commands

This set of commands represents a miscellaneous group that collectively tie together the system. All of the previous commands can be included in a macro string. Some of the following are also included in this class but many are best executed only as keyboard instructions.

3.8.1 L (length) - This command is used to specify the size of a block and should be used only as a keyboard command. The value which is entered is the exponent of the power of 2 and related to the number of bins in the real or imaginary halves (EG. If L=3, 2^L=8 or there will be 8 bins for the real part and 8 bins for the imaginary part of all three blocks). When this command is typed the computer responds by telling you the previous value (the power of 2) and an evaluation of this power of 2 in parenthesis (EG. If the previous

entry was 3, the computer responds by telling you 3 and (8)). After the computer's response the operator is expected to enter a new value and then a return key. This command should always be the first command executed at the start of any day's work since many interval constants are utilized through this operation. The maximum L is a function of the available memory (currently L8 for an 8K memory).

3.8.2 P (n) (Parameter)- This command is used to set a parameter and should also be used only as a keyboard instruction. The parameter is set by typing P and the parameter number. The computer responds by prompting the operator as to the nature of the parameter. The operator responds by entering the new parameter and typing the return key. Following is a list of the parameters:

P1 - This parameter is used with the "B" command to specify the number of itterations.

P2 - This parameter is used with the "D" command to relate the bin number with real time. The period of the digitizing clock should be stored as the parameter. Only integer values are valid.

P3 - This constant is used with the "S" (Shift) command and is referred to as the vertical shift constant.

P4 - This parameter is also used with the "D" command and is used to determine the interval over which an average is determined before an inflection point is evaluated.

P5 - This parameter is used by the N1 command and is the critical threshold value used by this program.

P6 - This parameter is used by the N2 command and is the constant which all amplitudes are divided by before the bin number (Block 2-real) is incremented.

P7 - Not Used

P8 - Not Used

P9 - This parameter is used as a file lock on entering a new macro string and storing a string. The key to unlocking the protection is to type an escape key after requesting P9 parameter modifications. Typing any other characters locks the protective feature. For security no prompt is typed and no characters are echoed back to the teleprinter during this operation.

3.8.3 W (constant) - This command allows constant data to be entered into a block. This should also be used as a strictly keyboard command, and not included into a macro string. Any single value can be entered into as many bins as the user desires. In this case the imaginary bins are considered a continuation of the real and , therefore, start at $2^L + 1$ (where L was the block specification under the L command.)

In practice the user types W. The computer asks for the constant value. The user enters this value and types return. The computer asks "From", the user types the first location the constant should appear and then types return. The computer types "TO", the user types the last bin number that the data should include through and then types return. If no logical errors occur (EG. To < From) the constant is stored and the message "OK" is typed. If a logical error occurs no message is typed and no data is stored. This command is useful in creating a data mask so as to eliminate undesired data, as well as other purposes.

3.8.4 M (Macro) - This command allows the operator to create a new macro string provided this function is not protected (See P9). A macro string is

a list of instructions up to 100 figures (letters and numbers) in length. The purpose for creating a string is to group together a number of instructions with attributes that the computer can execute in a sequential manner. This is the real benefit of CATSA since a string can be created quickly that allows one user to perform one task while another user may use CATSA for an entirely different kind of processing technique. Also a user can quickly change his string to allow for changes in his philosophy in a manner of seconds thus maintaining continuity with the problems of the minute.

When entering a macro string if a wrong character is typed the rub out key may be used. With every depression of this key the last character in the string will be eliminated and a "back-arrow" will be typed to tell the operator how many times he has struck the key.

Termination of a macro command simply requires typing another M. The macro command thus entered resides in an area referred to as the execution area. This command is obviously just a keyboard command.

3.8.5 R (Run) - Typing this character causes the macro string residing in the execution area to be acted on. The first command will be executed and upon completion, the second command, and so on. Execution will continue until either an attribute error is experienced or the last instruction has been executed.

3.8.6 Q (n) or QQ (n) (Queue) - Utilization of this command allows up to 9 macro commands to be stored away for use at a later time or to be brought to the execution area. If the first (and only) attribute is between 1 and 9 this implies that a macro string is to be brough from storage area n (value of attribute) to the execution area. If the first attribute is a

0 and the second attribute is 1-9 then a macro string is to be stored into area n from the execution area. Note that in both cases a copy is made from the original location and the original is left unchanged. If both attributes are 0 zero then the macro string in residence in the execution area will be typed on the console typewriter.

Summary:		
	QON	Execution area ⟶ Storage N
	QOO	Execution area ⟶ typewriter
	Qn	Storage n ⟶ execution area

3.8.7 B (n) (Begin again) - This instruction, when inserted into a macro string causes the flow of the execution of the commands to be altered. This command essentially restarts execution from the beginning 2^n times, where the range of n is 1→9. When n=0 string restart occurs the number of times specified in parameter P1. It is often useful to use B0 especially when the number of desired itterations is undetermined since any other specification would require rewriting the entire macro string just to change this string attribute.

3.8.8 Z (Define Restart) - This command is useful only with the previously described B command. This command is inserted between the beginning of the string and the B command. When this is done execution of the B command then restarts not at the beginning of the string but at the first instruction after the Z. Normally this command would follow the initial set up procedures such as E and V0.

3.8.9 U (n) (Shift) - This command effects a right circular shift of the data in block 1 real. This attribute ranges from 0 ⟶ 9 and except for 0 specifies the number of places to the right that the data will be shifted. Every time this command is executed the data is shifted the number of times it has been shifted in the past plus the value of the attribute. U0 clears

the counter that remembers the number of past shifts. With every shift a vertical bias is also added to the data, this value of bias is stored as parameter P3. This command is most useful if a great deal of data is to be displayed on a CRT and only several prominent peaks are in the data. (See Appendix for Example)

3.8.10 O (n) - This command executes user defined routines. Up to nine routines can be created and the specific routine is requested as the attribute. The valid range for the attribute is 1→9. Details for specifying a routine are explained in Appendix B.

Conclusion

The objective of this thesis was simple: Create a computer system that utilizes programming which is not time consuming to use or learn by the researcher. The result is a multiple input computer system with software that functions to perform large data manipulations with a few powerful instructions. The operator can execute individual instructions to mold his data into a uniquely useable form. The operator can also string together a number of instructions and have these performed sequentially when the command R is typed. In this latter case the researcher builds a processing routine or program from a number of basic instructions.

This system is presently being successfully used by several researchers in such studies as convergent eye movement, visual evoked cortical potential, and optokinetic nystagmus. Having a single system that will handle these divergent tasks increases the cost effectiveness of the equipment while providing many researchers with a valuable tool.

Bibliography

1. Data General Corporation, How to Use the Nova Computers, Data General Corporation, Southboro, Massachusetts, April, 1976

2. Gold, B. and C. Rader, Digital Processing of Signals, McGraw-Hill Book Company, New York, 1969

3. Kreyszig, Erwin, Advanced Engineering Mathematics, John Wiley and Sons, Inc., March, 1964

4. Otnes, R. and L. Enochson, Digital Time Series Analysis, John Wiley and Sons, New York, 1972

5. Rabiner, L, and C. Rader, Digital Signal Processing, IEEE Press, New York, 1972

6. Regan, D., Evoked Potentials in Psychology, Sensory Physiology, and Clinical Medicine, Wiley-Interscience, New York, 1972

7. Texas Instruments, Inc., The TTL Data Book for Design Engineers, Texas Instruments, Inc., 1973

Appendix A

Command Summary

Function	Command	Attribute	BLK1	BLK 2	BLK 3
Absolute Value	A	none	blk 1		
Repeat P1 times	B	0			
2^n times		n=1-9			
Copy	C	2	1 re 1 im	1 re 1 im	
	C	3	1 re 1 im		1 re 1 im
Digital Display	D	0-9			
if 0 use P4 for Av.					
Expunge all memory	E		0 0	0 0	0 0
FFT	F		re im		
GET data	G	1	1		
	G	2	1	2	
	G	3	1	2	3
Swaps Re and Im parts	H	1	im re		
	H	2		im re	
	H	3			im re
Inverse FFT	I		+T -T		
Clears Blck 1 im	J		0		
Integrate Block 1	K		re		
Specifies block length	L				
Specify Macro string	M				
Normalize block 1 by 2^n	NO N=1-9		BLK $1/2^n$		
Acquire ISI	N1		ISI amp		
Histogram	N2			histo	
Change parameter N	P	N			
Type string	QO	O			
Recall string N	Q	N			
Store string at N	QO	N			
Run Macro string	R				
Swap blocks	S	2	2 re 2 im	1 re 1 im	
	S	3	3 re 3 im		1 re 1 im
Wait for trigger pulse	T				
Clear chift height	U	0			
Shift right circular	U	Times	ijabcdefgh		
Erase screen	VO				
Plot Rec.	V1				
Plot Polar	V2				
Plot zero reference	V3				
Hard copy plot	V4				
Create a constant in BK-1	W		constant		
Mult. BK 1=BK2xBK1	X		BK1 x BK2		
Scale real new max/min	Y1		scl		
Scale whole block 1 new max	Y2		scl scl		
Scale real old max/min	Y3		scl		
Scale whole block old max	Y4		scl scl		
Loop back from B	Z				
Add BLK3=BK1 + BK3	+				BK3 + BK1
Sub BLK3=BK3 - BK1	-				BK3 - BK1

P1 - used to specify loop cycles used with BO
P2 - used with D to specify time interval
P3 - vertical shift constant
P4 - averaging interval for DO
P5 - amp threshold w/N1
P6 - bin scale w/N2
P9 - security lock

Appendix B

Example Uses:

C.1 Visual Evoked Response (VER)

One widely used method to extract the VER from the background EEG (noise) is to time lock average. With this technique samples of the EEG start being acquired with the occurrence of the evoking stimulus. The new samples are summed with previously acquired samples such that all samples occurring after the given stimulus are added together bin by bin. This forms an ensemble average and since only the VER is time locked it will tend to grow larger with increasing numbers of stimuli while random events (non-time locked will grow slower with one another (data grows at N, noise at $\sqrt{N}$)

A macro string which will accomplish such an operation follows:

Instructions	Explanation
VO	(erase the CRT)
E	(erase all of memory)
Z	(loop return point)
T	(wait till a trigger pulse occurs)
G1	(Fill block 1 real with data)
+	(sum this data into block 3)
BO	(Loop back to Z the number of times specified in parameter P1)
S3	(swap data between blocks 1 and 3)
Y1	(amplitude scale block 1 real)
V1	(display block 1 real)
V3	(plot a zero reference line)
M	(end of Macro string)

This string will cause the system to acquire the number of data points specified under the L command, at a rate set by the external clock. P1 is set to the number of itterations to be performed, before executing the string. If the clock was set at 500 Hz, the time between samples will be 1/5000 or .002 sec (2 ms). If L is set to 5, ($2^5 = 32$) then the total input sample duration would represent 32 bins x 0.002 sec. or .064 sec (64 ms).

C.2 Analysis of the Frequency Components of a Wave Form

The fast Fourier transform is useful for determining the frequency components of any wave form. The following string will accomplish this:

VO	(erase the CRT)
E	(erase memory)
G1	(acquire the data)
F	(perform the FFT)
A	(Obtain the power spectrum of the FFT)
Y1	(amplitude scale the real portion of the FFT)
00	(smooth the power spectrum)
V1	(display the power spectrum)
V3	(plot a zero reference)
D1	(display the peaks in a digital form)
M	(end of Macro string)

The frequency resolution would be the frequency difference between coefficients or (250 Hz/16) 15.625. The sample represents 64 ms (bin size x sampling interval or 32 x 2 ms) worth of time. Obviously for greater resolution the block size need be increased or the clock frequency lowered

Because of truncation error or inexact frequency components and related phenomenon the center of a given frequency component may be difficult to locate even after smoothing. In this case it may be useful to integrate the power spectrum and locate the center with this technique. The following macro string will perform this function:

K	(integrate block 1, real)
Y1	(scale block 1, real)
V1	(display the integral)
M	(end of string)

If the clock rate was set at 500 Hz the last coefficient in the power spectrum would represent 250 Hz (1/2 x 500 Hz). If L was set at 5 (2^5 = 32) then 17 coefficients would be displayed (1/2 x bin size + 1). The first coefficient would represent the energy at 0 Hz (the DC component). The second coefficient would represent the energy at 15.625 Hz (250 Hz/16), the third coefficient would represent the energy at 31.25 Hz (2 x 250 Hz/16), the fourth represents 46.875 Hz (3 x 250 Hz/16), and so on. The time resolution would be 2 ms.

C.3 Digital Filtering

Often it is desirable to examine how a particular frequency band behaves as a function of time, exclusive of the rest of the spectrum. This requires a high quality filter and can be easily implemented on a digital system through the use of the FFT. The general technique is to convert a signal into the frequency domain, remove all but the desired frequency components, and then revert the signal back into the time domain. The following Macro String accomplishes this:

VO	(erase the CRT)
J	(erase imaginary part of block 1)
G1	(acquire the data)
F	(convert the data into the frequency domain)

X	(remove the undesired components by multiplying Block 2 x Block 1)
I	(revert back to the time domain)
Y1	(amplitude scale the results)
V1	(display the data)
V3	(plot a zero reference)
M	(end of string)

The key to this process is really the fifth step, multiplying Block 2 times Block 1. Block 2 has to contain a constant which was created using the "W" command. This process is best explained by example as follows: Assume that L was set to 5, then the block size is 32 (2^5). This means that 32 bins are reserved for the real part of block 1, and 32 for the imaginary part of block 1. When the FFT is performed more points in the block exist than the FFT can produce without redundancy. Bin 1 contains the DC component, Bin 2 and 32 contain the same value, Bin 3 and 31 have the same value and so on. This kind of scheme is also true for the imaginary part, bin 33 contains the DC component, bins 34 and 64 have the same magnitude (but opposite signs), bin 35 and 63 have the same magnitude and so on.

Assume that the clock rate is 500 Hz. This means that each coefficient will be a multiple of 250/16 Hz or 15.625 Hz. Further, assume we are interested only in the frequency range of 10 Hz-35 Hz. This means we would like to eliminate all other components. This is accomplished by setting all coefficients to zero except those associated with that range. This means that bins 1 and 2 sould be among those of interest. Remembering not to ignore the redundancy or the imaginary part then the following bins would be of interest: 1, 2, 26, 27, 34, 35, 58, and 59. All other bins should be set at zero.

This is accomplished by first erasing memory with "E" then using W to create "ones" wherever we want data to remain untouched:

W	AMP 1	FROM 1 to 2
W	AMP 1	FROM 26 to 27
W	AMP 1	FROM 34 to 35
W	AMP 1	FROM 58 to 59

This "filter" is then copied to block 2 using "C2" and the Macro string is ready to be run!

C.4 Histogram display of ISI acquired data

One method of obtaining frequency domain information when only one frequency component is present is by examining the time between cycle and displaying the results as a statistical histogram. The following Macro string will serve to accomplish this task:

V0	(erase CRT)
E	(erase memory)
N1	(acquire data in ISI mode)
N2	(form a histogram of the data from block 1 real into block 2)
S2	(place the histogram into block 1)
Y1	(amplitude scale the data)
V1	(display block 1 on CRT)
V3	(plot z zero reference)
M	(end of string)

Since the peak amplitude between each ISI period is stored in the imaginary half of block 1, a histogram of this data may be useful. The following Macro string can be used to obtain this display. This string assumes that the previous Macro string has just run.

J	(clear the imaginary half of Blk 1)
H1	(swap the real and imaginary halves of Blk 1, this essentially clears Blk 1 real)
S2	(bring the ISI data back into block 1)

H1	(place the amplitude values into the real part)
N2	(obtain the histogram)
S2	(bring histogram into block 1)
Y1V1V3	(scale and display data with a zero reference)
M	(end of string)

C.5 Average Two Channels of Information Simultaneously

This process is somewhat round about but possible with the following string:

VOE	(Clear CRT and memory)
Z	(loop back point)
T	(wait for trigger pulse)
G2	(acquire 2 channels of data, one in block 1 real and one in block 2 real)
+	(add block 1 real into block 3 real)
S2	(swap blocks 1 with 2)
H1	(place data into imaginary 1/2)
+	(add this data into block 3)
J	(clear imaginary part of block 1)
H2	(place clear part of block 2 into imaginary part)
BO	(return to beginning P1 times)
S3	(place averaged data into block 1)
Y1V1	(scale and plot the real part)
H1Y1V1	(scale and plot the imaginary part)
M	(end of string)

Appendix C

Technical Description of CATSA

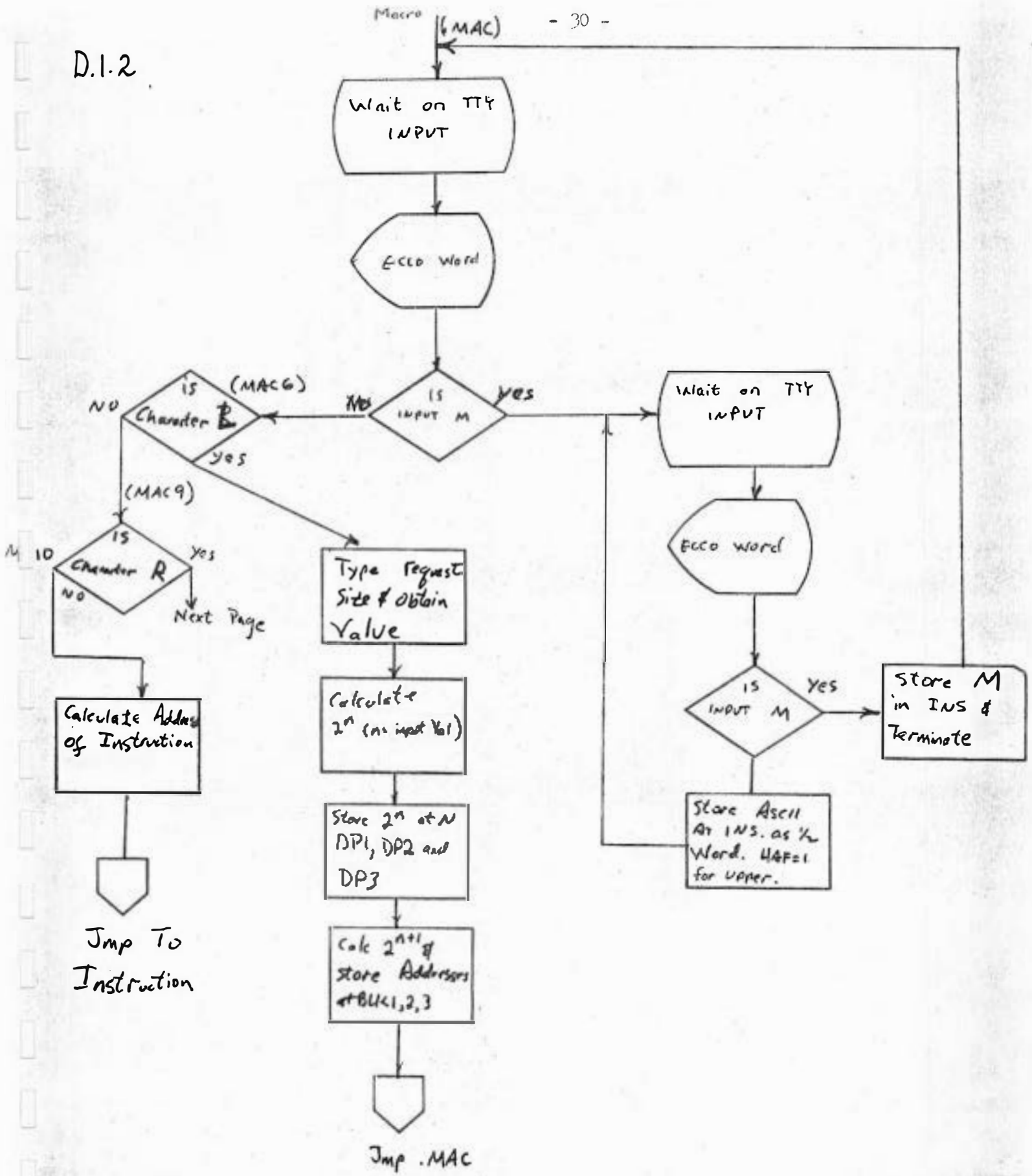
D.1 Program Entry Point

Location 50_8 contains an instruction to jump to the relocatable symbol MACRO the actual entry point. In effect then the absolute address 50_8 serves as one entry point and the symbolic entry point is MACRO.

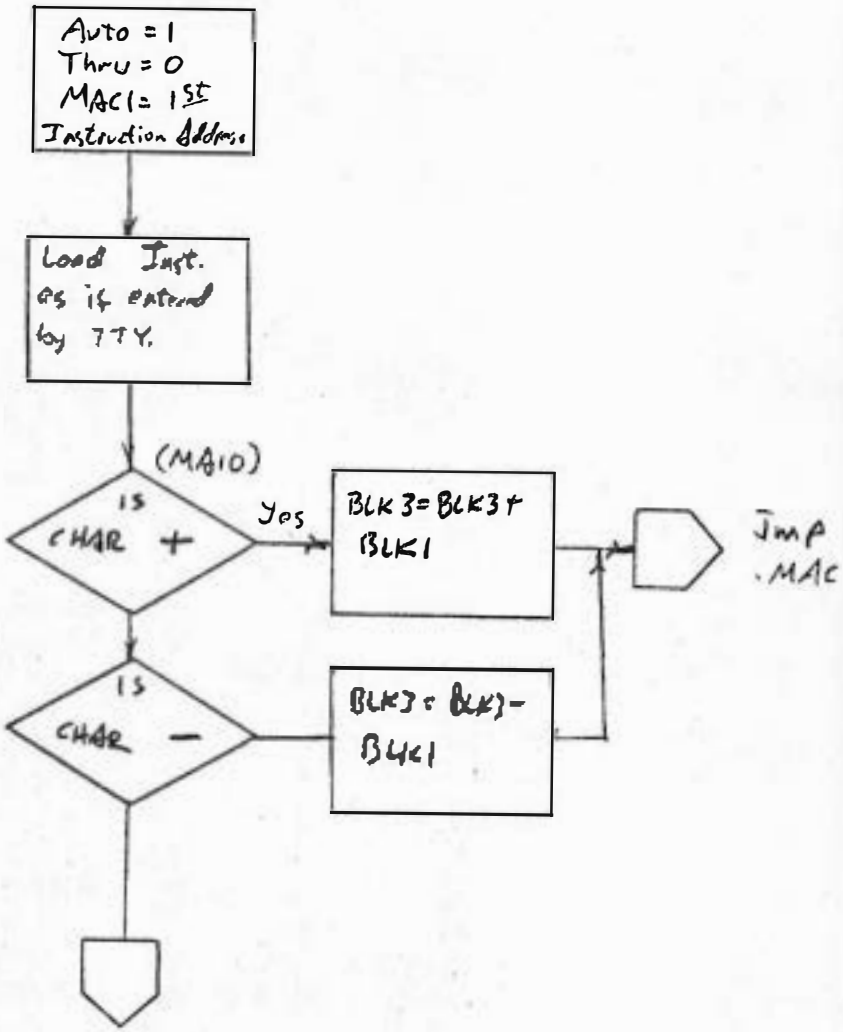
D.1.1 Entry Initialization

Upon entry at MACRO a teletype carriage return is executed and a question mark is typed. The flow chart on the next page explains the remainder of the operations. In addition to this chart a number of charts and explanations detailing the remainder of the program follows. In general only those routines that will aid the user create his own optional programs will be discussed in detail. For those not discussed refer to the program listing (Appendix F).

D.1.2

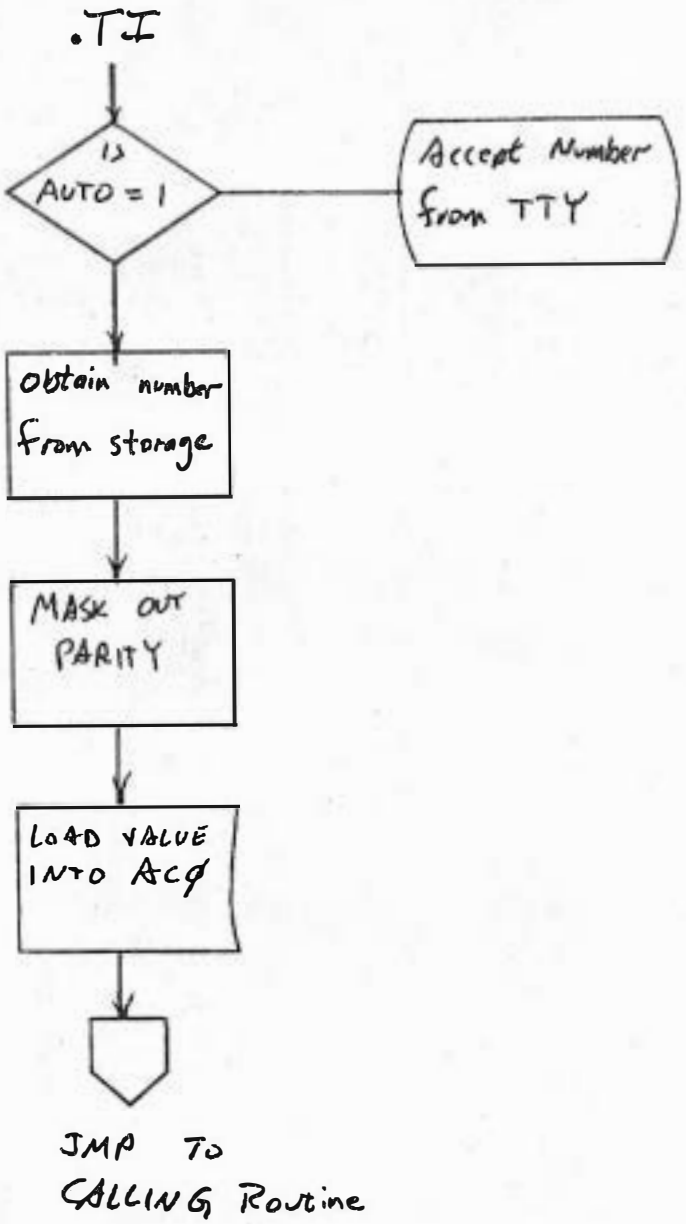

Macro (MAC)
Wait on TTY INPUT
Ecco Word
Is INPUT M
Yes
No
Is Character R
(MAC6)
NO
yes
(MAC9)
Is Character R
NO
10
yes
Next Page
Calculate Address of Instruction
Jmp To Instruction
Type request Site # Obtain Value
Calculate 2^n (n= input Val)
Store 2^n at N DP1, DP2 and DP3
Calc 2^n+1 # Store Addresses of BUF1,2,3
Jmp .MAC
Wait on TTY INPUT
Ecco Word
Is INPUT M
Yes
Store M in INS & Terminate
Store Ascii At INS. as 1/2 Word. HAF=1 for upper.

Character is an R:



D.2

Const. input Subroutine



D.3 LIM checks the upper and lower limits of a constant for proper boundry conditions.

Sequence: JSR LIM
ASCII Lower Limit
ASCII Upper Limit
Address of error message
Normal Return

If an out of bound condition existed the error message specified is typed whether the system is in automatic mode or not.

D.4 Out-Text Output Subroutine

Calling Sequence: JSR @OUT
Address of message
Normal Return

Special considerations: The carriage routine CRLF counts 50 characters before actually outputting a carriage return. One can force a carriage return by using this subroutine (OUT) by JSR'ing to .OUC.
All data is assumed stored in a .Tx+M1 format and a null character terminates the output (forced by< 0 > at the end of the text string)
Return of this subroutine is to the calling program +2.

D.5 BEGIN This is an instruction program and will cause the program to restart to the beginning or Z the number of times specified by the attribute,a P5 (if the attribute is zero).

Attribute Limit ASCII: 60 71

error message is BEG: "BEGIN"

Subroutine EXP is used to calculate the number of itterations as a power of two of the attributes

Symbols

THRU: Number of itterations, counter, this is incremented and tested every pass. If the value is less than the required number of itterations return is made to RET after resetting the program location counter to TER 1. (TER 1= the first location or the address of Z if it was encountered).

D.6 ADPL, ADMI Add and Subtract Routines

ADSI is set to 1 for addition and 0 for subtraction

ADS2 is a loop counter to count the number of elements in a block

This value is initialized to the value of BKS (set to $2^n + 1$ by command L where n is the input value). This value is decremented and tested after every operation by a DSZ command. DP3 is set to the value of DP 1 at the end of this operation

D.7 RET - Return Point for all Major Subroutines

JMP @ RET is the correct return instruction for all command routines. This routine compares AUTO with 1 and if it is equal to this value pulls the next ASCII word from the execution area and returns to the decoding section. If AUTO is not 1 it returns to MACRO which will in turn type a question mark and await the next command.

D.8 YSCL - Amplitude Scaling Routine

This program searches block 1 for the extreme maximum and minimum values and stores these at SCL 2 (minimum value) and SCL 3 (maximum value). The values are then linearly scaled to a minimum of zero and a maximum of 1023_{10} using the linear regression formula $y = mx + b$. Calculations are handled as double precision variables and the slope "m" is calculated as needed to avoid truncation error problems.

D.9 BIND - Binary to Digital Output Routine

This routine accepts words in AC0, converts it to digital (ASCII) and passes the output to PUT to be typed on the console typewriter (if GNUM=1) or on the CRT (if GNUM=0). Leading zeros and the + sign are suppressed on positive numbers and a minus sign is typed on negative numbers (indicated by Bit 0=1).

D.10 PLOTT - CRT Number Generator

This program accepts ASCII numbers in AC0 and plots them on the CRT. Non valid ASCII characters are plotted as E (error), a minus sign is also a valid character. The character is plotted from an initial X and Y coordinate SC and SY and is based on a first coordinate scheme with maximum values of 0 to 1023_{10}. The size of the character is specified in SPAC. The ASCII character is stored in CHAR. At the terminus of this subroutine a jump is made to the address specified at the entering AC3 address (normal subroutine return). The value of SX is properly incremented just prior to this return so that the next character will be spaced well.

This routine uses the conventional seven segment code in a look-up table format at location KEY.

D.11 .DBIN - accepts TTY Digital Numbers and Converts it to Binary

This subroutine accepts a digital number from the console typewriter, converts it to binary, and passes it back to the calling routine through AC1. GET is used by this routine to acquire the teletype parameter and may be used independently to acquire TTY information (information is returned in AC0; all characters but carriage return are echoed; NEW is incremented with each echo).

Appendix D

Adding Additional Subroutines

General

Nine areas have been set aside for small user added subroutines (written in Nova 800 assembly language). Each of these areas has been initialized as a loop to cycle BKS times (double the block size). All accumulators are free to the user and unless expanded 30_8 (octal) words are available for his program. The user's program is accessed through command 0 (option) with attributes 1-9 being valid and specific for each program.

Details (Refer to page 469 Appendix F) The symbolic entry point to each routine is OP1 through OP9. OPT1 is initialized to BKS (2^{n+1} as set by the L command where the n= the entered number). Before this value is stored a NO-OP (JMP .+1) instruction is encountered so that if the user decides to treat only 2^n bins a MOVSR 1, 1 instruction should replace this NO-OP instruction. After successful completion of the loop a systems return is accomplished to symbolic location RET. Note that if not all of the 30 locations are used no JMP is neededsince the spaces are filled with NO-OP instructions. For convenience a counter (OPT2) is automatically initialized to zero and incremented after every pass.

Hints Block starting addresses are Blk 1,Blk 2, Blk 3. The first address of the real segment is, therefore, Blk 1 (for block 1 and Blk 1 + n is the imaginary starting address.

Example Suppose we wished to accept an attribute and add this number to all locations in Block 1. Here is the program that would be inserted into the NO-OP portion of the option area

```
JSR @ TI        This subroutine gets the attribute and stores it
                at AC0.

LDA 1, Blk 1    Get the address of block 1.
```

```
LDA 2, OPT 2      Get the automatic incrementor
ADD 1, 2          Compute the present bin's address
LDA 3, 0, 2       Load the value of the computed bin into AC3
ADD 0, 3          Add in the value of the accepted parameter
STA 3, 0, 2       Store this new value back into memory
```

Note that as the option is set up the computer would expect an attribute for each pass. To avoid this problem the "loop-back" address has to be incremented by one. This instruction is located just after the "DSZ OPT 1" instruction and should be changed to "JMP . -30".

This routine would be executed by typing "01N" where N would be the value added to all the bins in block 1. At the completion of this task the next question mark would be typed or the next instruction of a macro string would be executed.

Appendix E

Computer Interface

This circuit diagram represents the efforts of Dennis Engdahl, a former Pacific University employee and myself. It was specifically designed for this anlysis system and ties together all of the peripheral equipment (D/A and A/D converters) with the Nova 800 computer. It is included as an appendix to this software document since the detailed inner workings of the software are impossible without exact knowledge of the peripheral devices.

Following is a description of the instructions that specifically relate to this interface:

Digital to Analog Converter Instructions:

Device Code: 23 (following the Nova convention)

DOC: Set scope mode according to data bits 13-15

If bit is 1 it conveys the following meaning:

Bit 13 = VIEW

Bit 14 = WRITE THROUGH

Bit 15 = NON-STORE

DIB: Scope status is read into accumulator.

Bit 15 = 1 indicates that Erase is still occurring.

DOA: Bits 0 and 6-15 of the data from the X coordinate

DOB: Bits 0 and 6-15 of the data from the Y coordinate

Control modifiers (Bits 8 and 9 of instruction)

S intensifies beam and pulses converter

P erases the screen

C pulses the converter

Analog to Digital Converter Instructions:

DOA - Loads the "channel select" latch. These data bits have the following meaning:

Bit 12	0 = conversion starts on command
	1 = conversion starts after first receiving a command and then receiving an external trigger pulse.
Bits 13-5	Channel number 0-7

DIC - Read the converted input Bits 0, 4-15 are used

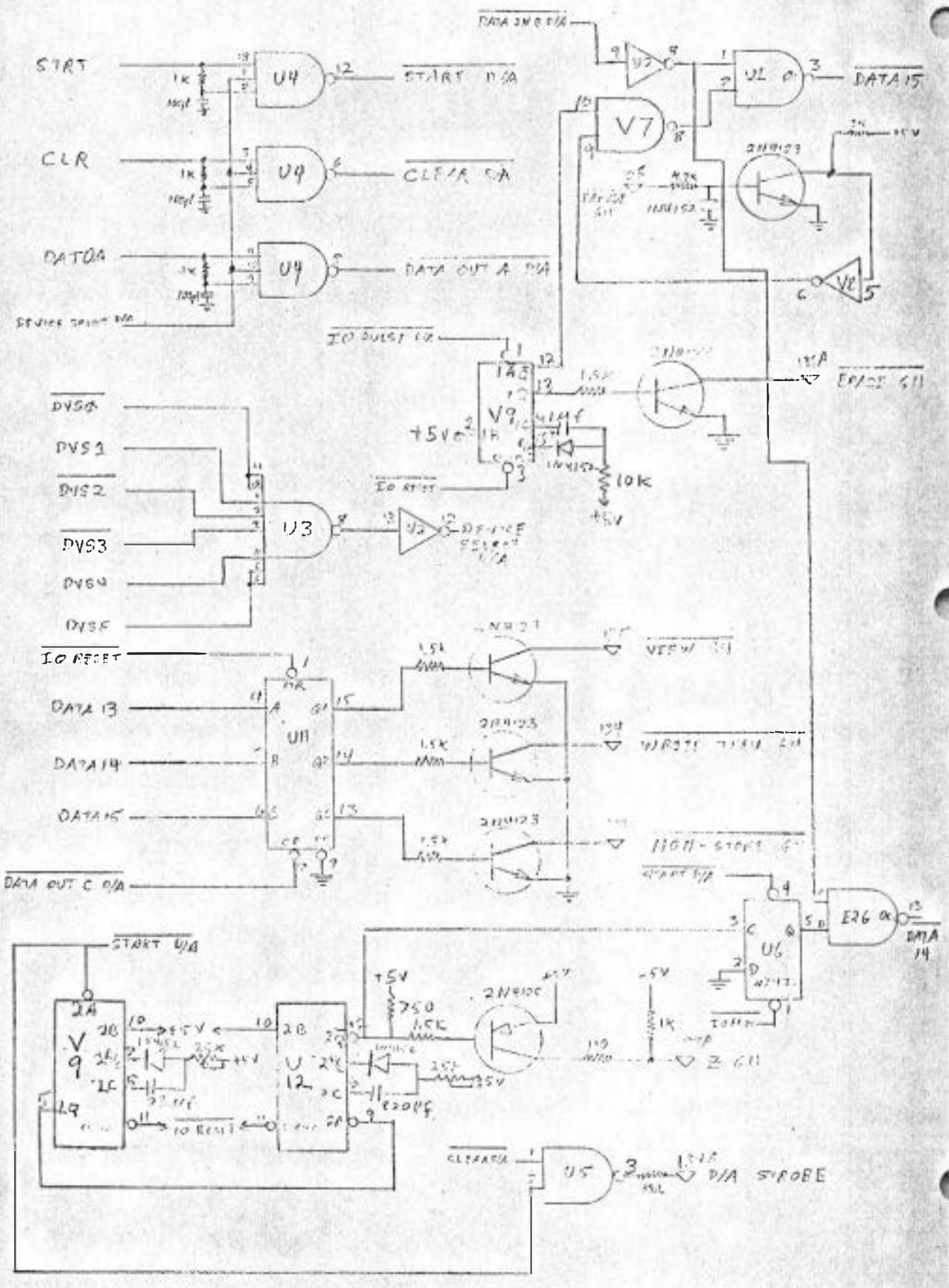
DIA - Read the "channel select" latch, observe that only bits 13-15 are read.

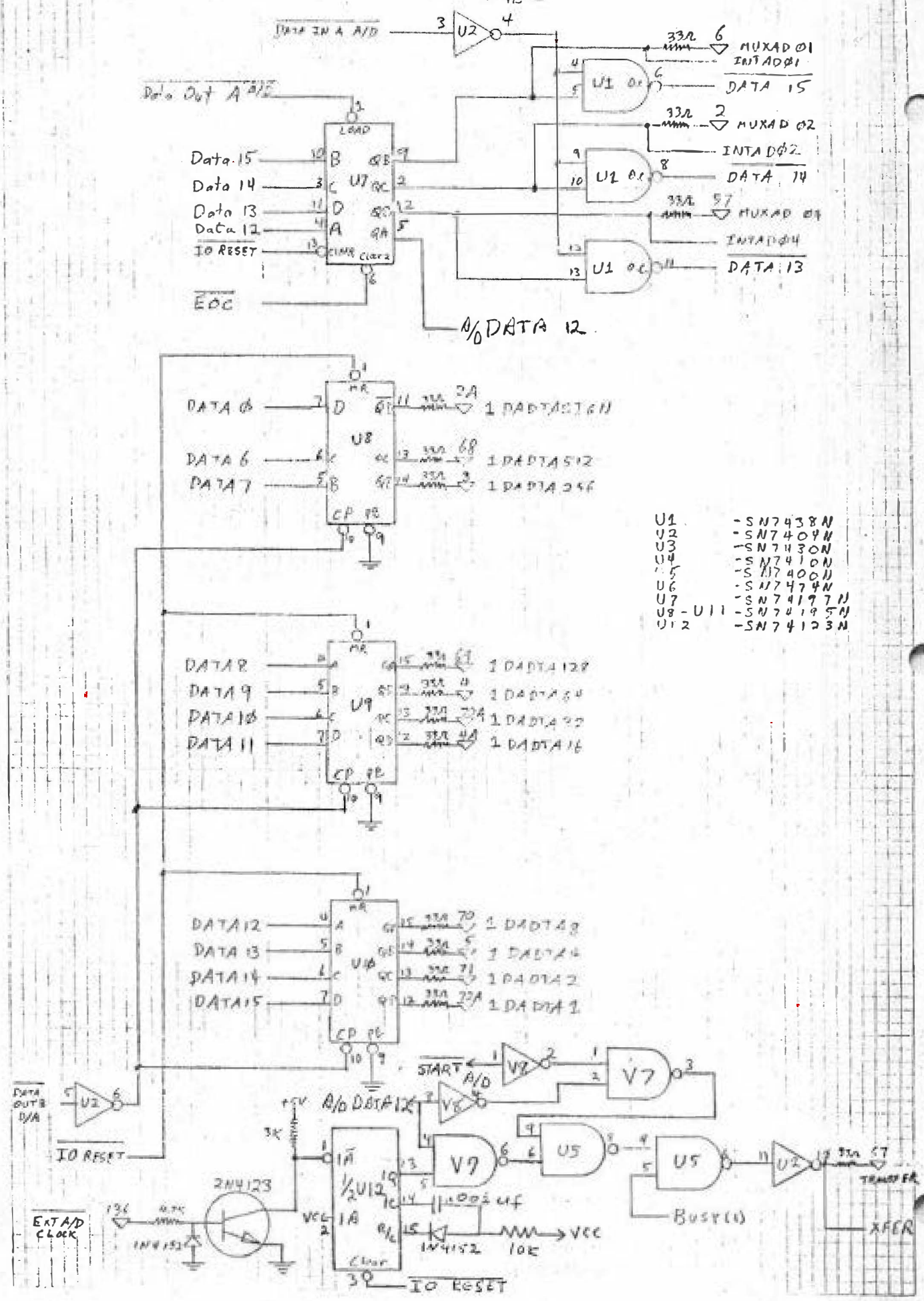
NOTE: The channel select value is incremented automatically after every conversion.

Control modifiers

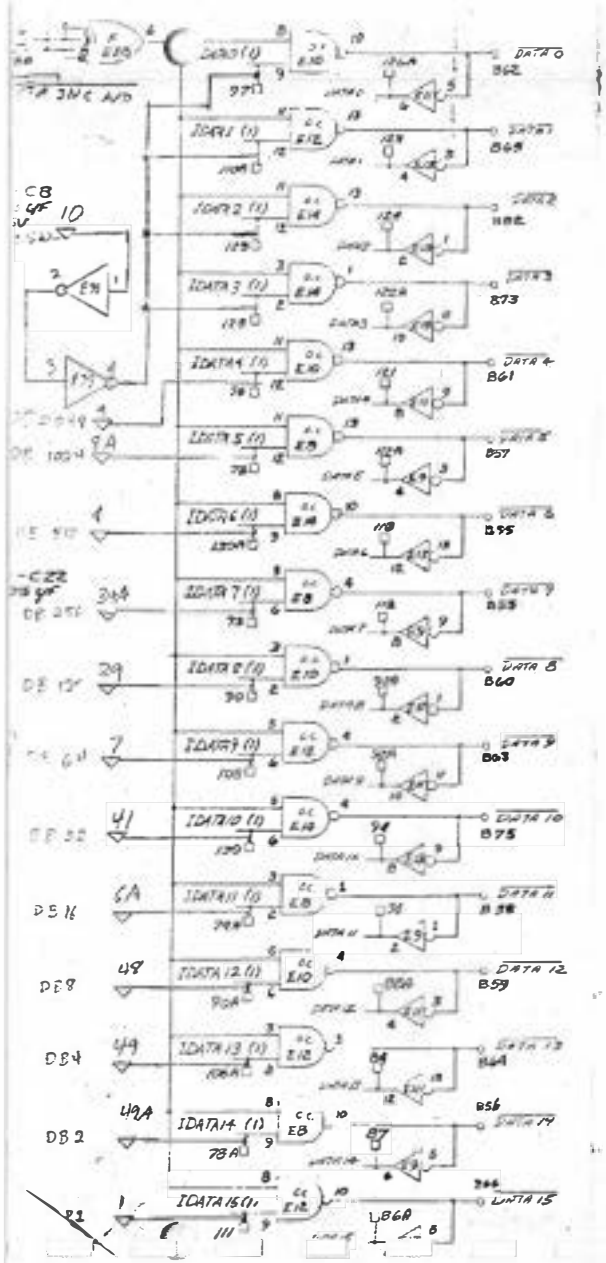
S - commands the start of conversion

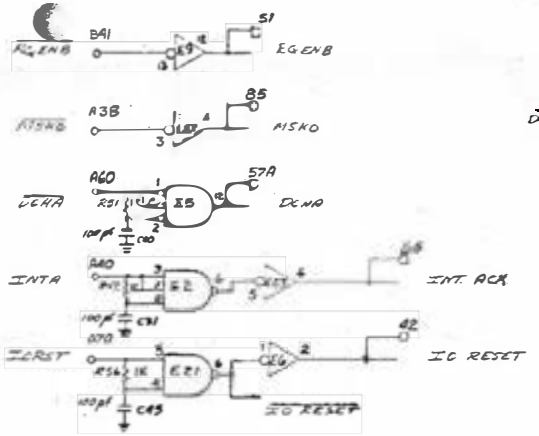
(Note: The DOA bit 12 status affects the actual start of conversion)

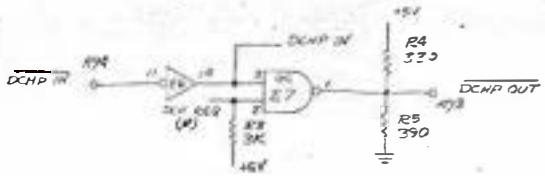

START
CLR
DATOA
U4
U9
U4
START D/A
CLEAR D/A
DATA OUT A D/A
DATA 15
V7
V8
V9
IO PULSE (2)
IO RESET
+5V
10K
DVS0
DVS1
DVS2
DVS3
DVS4
DVSF
U3
DATA 13
DATA 14
DATA 15
U11
DATA OUT C D/A
VERIFY (1)
WRITE (1)
HIGH - STOP
START D/A
U6
E26
DATA 14
START D/A
V9
U12
+5V
2N4403
-5V
1K
10 RESET
CLEAR D/A
U5
D/A STROBE

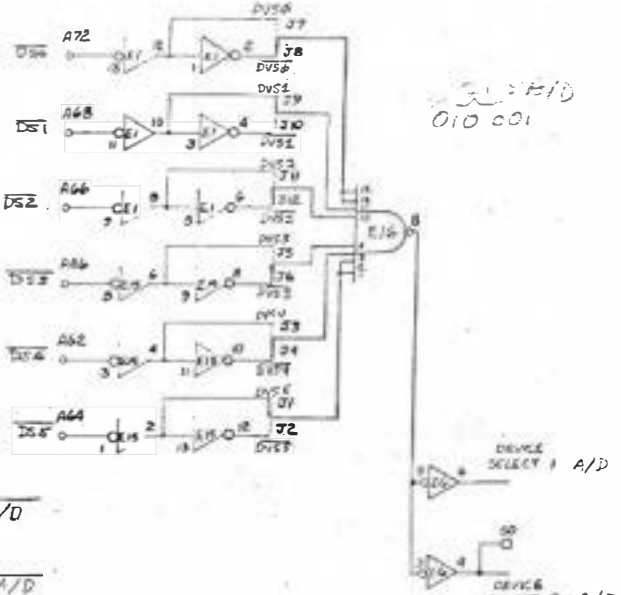


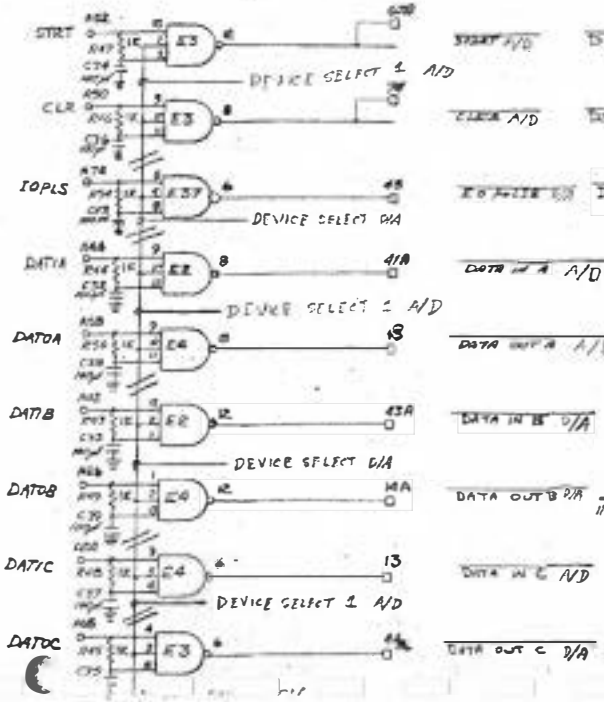
U1	- SN7438N
U2	- SN7404N
U3	- SN7430N
U4	- SN7410N
U5	- SN7400N
U6	- SN7474N
U7	- SN74197N
U8 - U11	- SN74195N
U12	- SN74123N

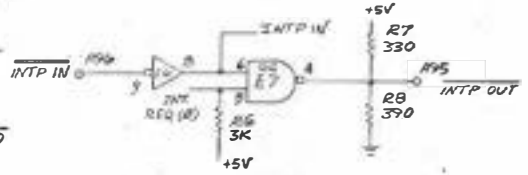

IDATA 0 (I)
IDATA 1 (I)
IDATA 2 (I)
IDATA 3 (I)
IDATA 4 (I)
IDATA 5 (I)
IDATA 6 (I)
IDATA 7 (I)
IDATA 8 (I)
IDATA 9 (I)
IDATA 10 (I)
IDATA 11 (I)
IDATA 12 (I)
IDATA 13 (I)
IDATA 14 (I)
IDATA 15 (I)
DATA 0
DATA 1
DATA 2
DATA 3
DATA 4
DATA 5
DATA 6
DATA 7
DATA 8
DATA 9
DATA 10
DATA 11
DATA 12
DATA 13
DATA 14
DATA 15


EGENB
MISKO
DCMA
INTA
INT. ACK
IO RESET


DCMP IN
DCMP OUT
+5V
R4 330
R5 390
3K
+5V

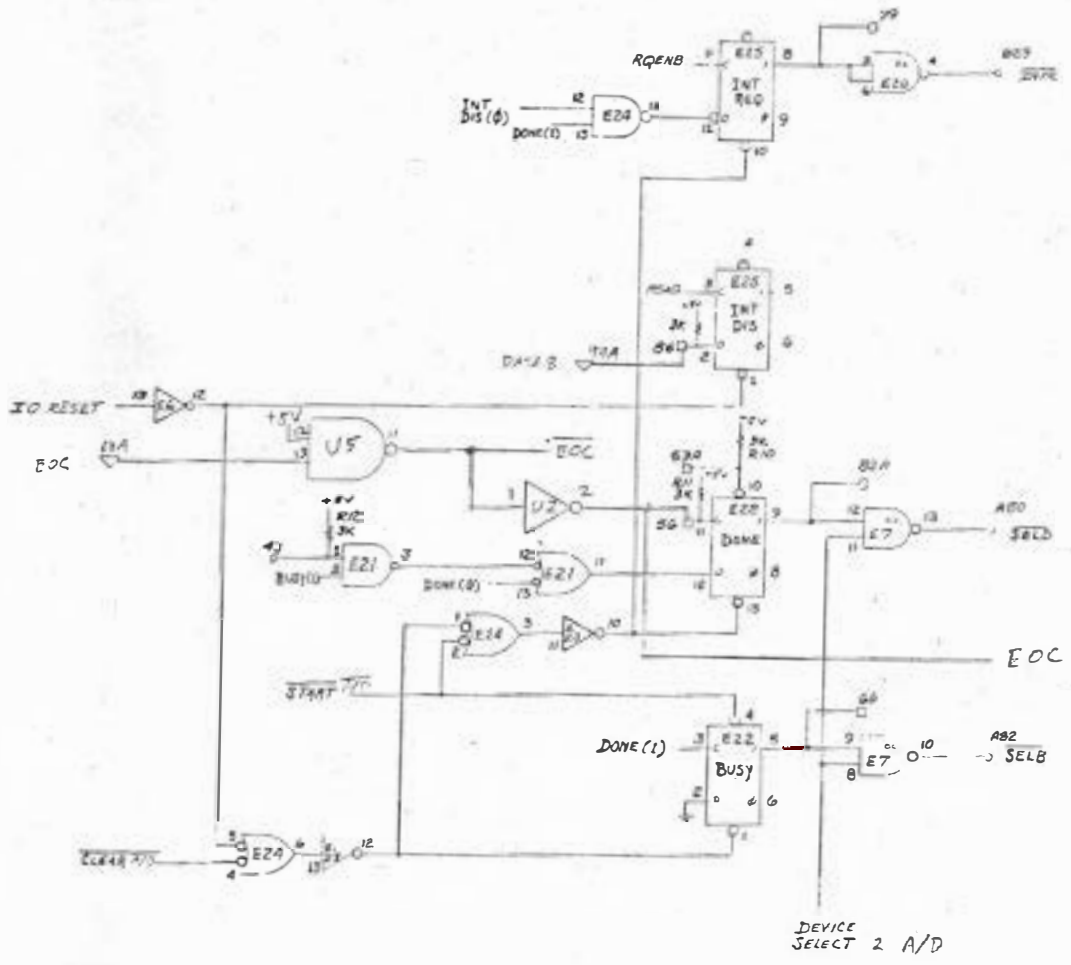

DVS0
DVS1
DVS2
DVS3
DVS4
DVS5
DEVICE SELECT 1 A/D
DEVICE SELECT 2 A/D

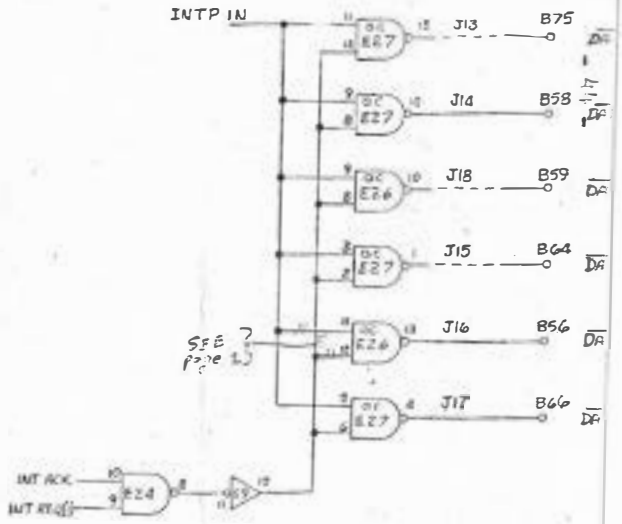

STRT
CLR
IOPLS
DATA
DATOA
DATOB
DATOC
DEVICE SELECT 1 A/D
DEVICE SELECT D/A
DEVICE SELECT 2 A/D
START A/D
CLEAR A/D
DATA IN A A/D
DATA IN B D/A
DATA OUT B D/A
DATA IN C A/D
DATA OUT C D/A


INTP IN
INTP OUT
+5V
R7 330
R8 390
R6 3K
+5V

INS AVAILABLE FOR INTERFACE

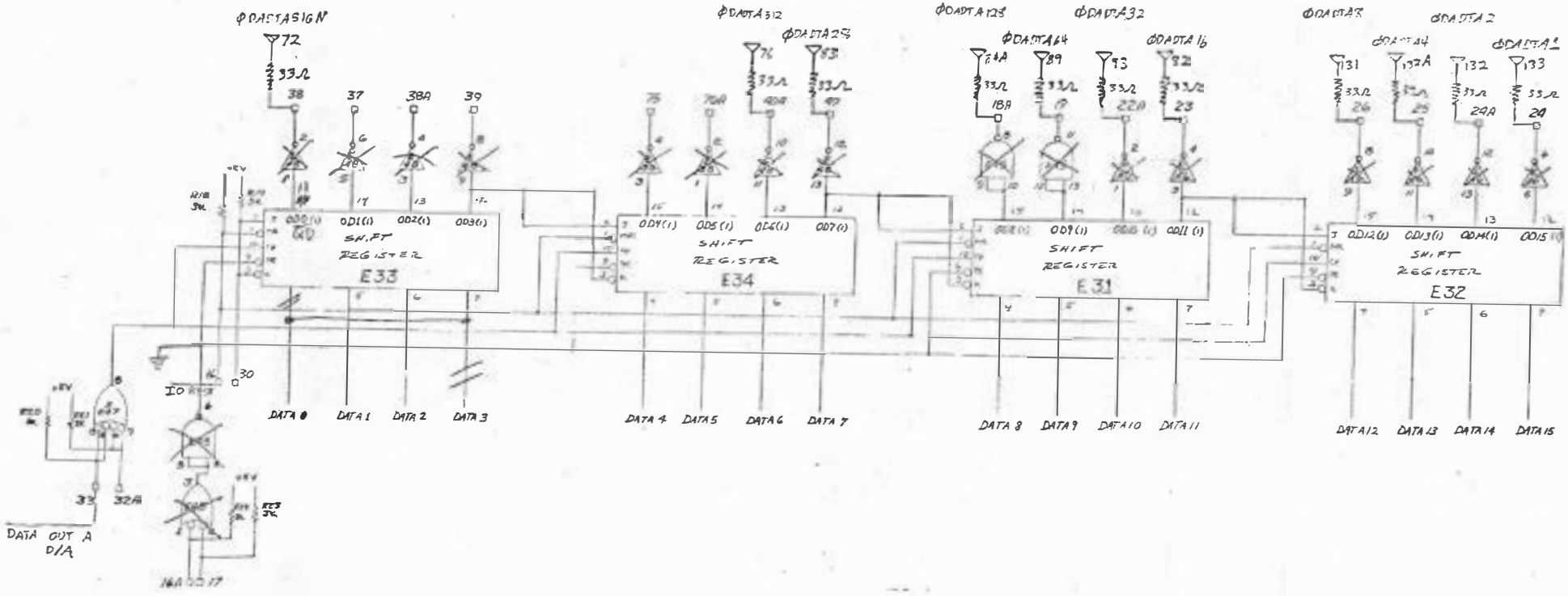
A47	10
A49	9
A57	8A
A59	8
A61	34A
A63	29
A65	7
A67	41
A69	6A
A71	48
A73	49
A75	49A
A76	1
A77	6
A78	2
A79	57
A81	67
A83	68A
A84	2A
A85	68
A86	3
A87	69
A89	4
A89	70A
A90	4A
A91	70
A92	5
B6	71
B11	72A
B13	72
B15	76
B19	83
B23	84A
B25	89
B27	93
B31	82
B34	131
B36	132A
B38	132
B40	133
B48	134A
B49	98A
B51	104
B52	134
B53	135
B54	136A
B47	135





NOTE 1 □ DESIGNATES WIRE WRAP PIN

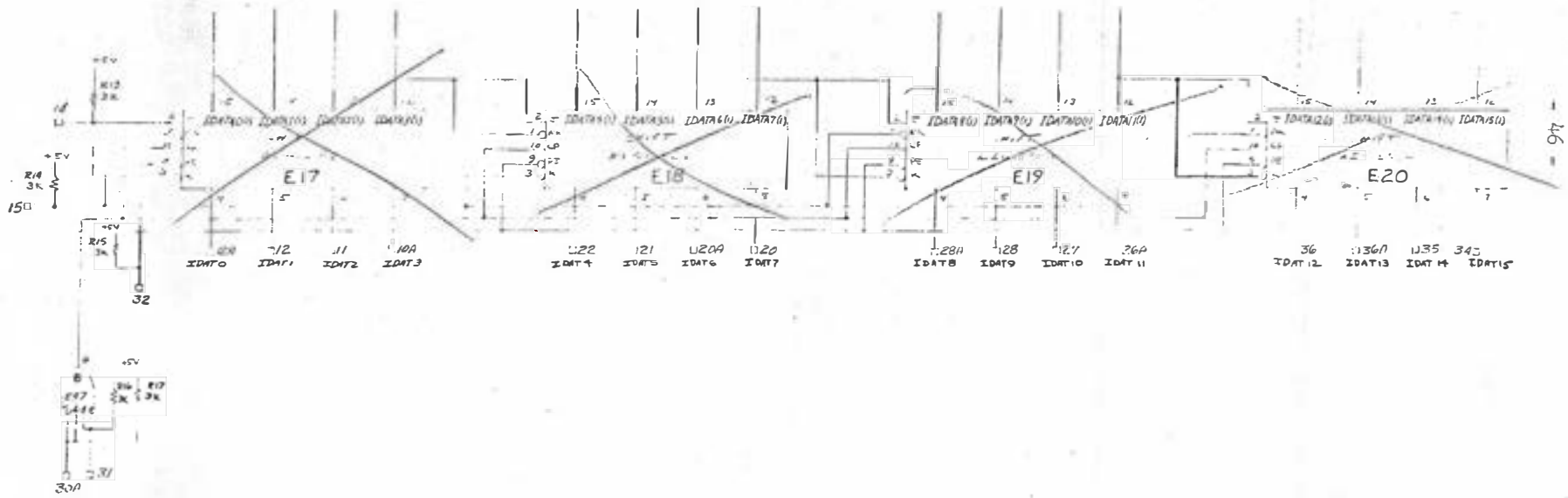
○ DESIGNATES EDGE CARD CONN.



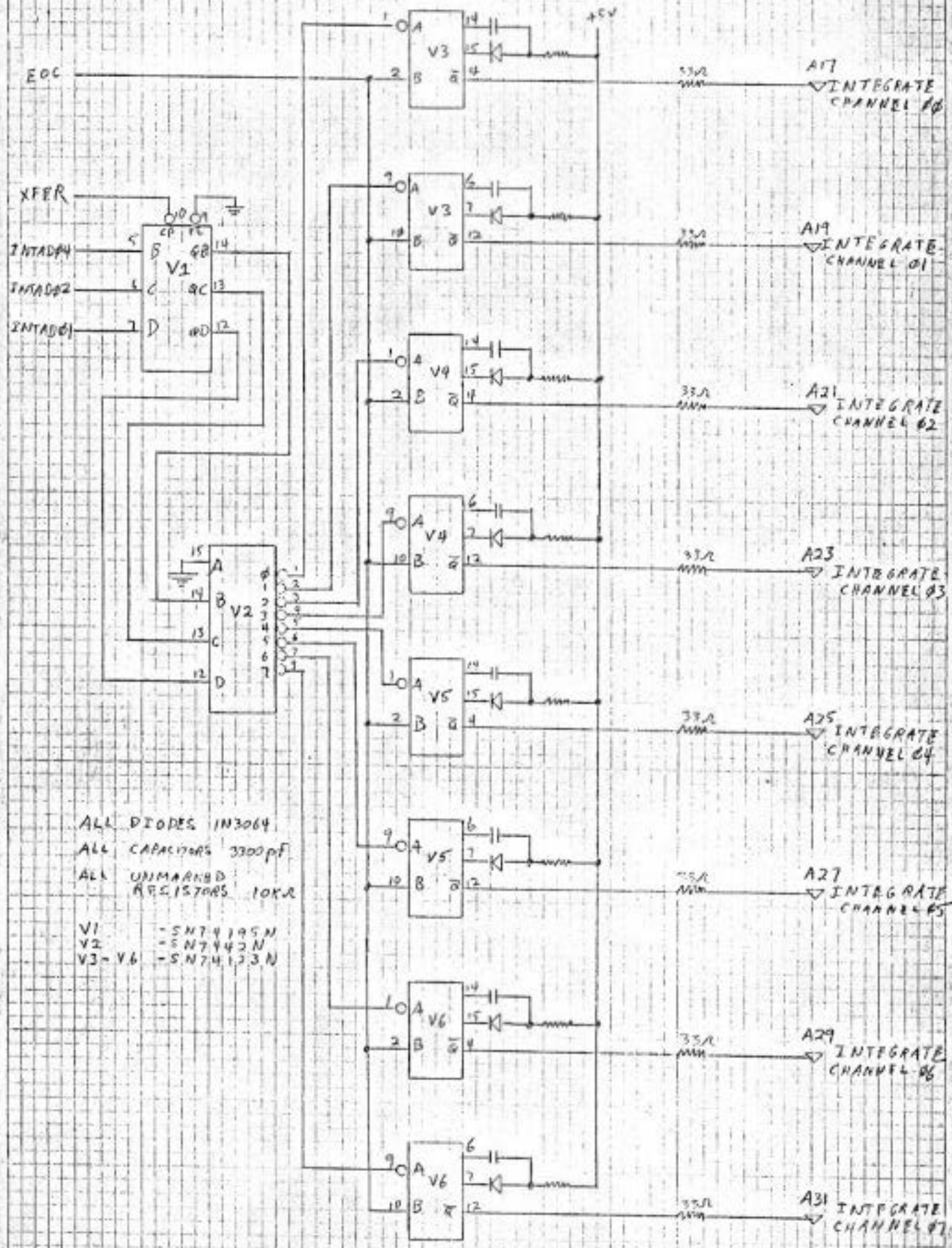
NOTE 1. □ DESIGNATES WIRE WRAP PIN
○ DESIGNATES EDGE CARD CONN.

OUTPUT REGISTER

D/A, A/D INTERFACE



NOTE 1 □ DESIGNATES WIRE WRAP PIN
○ DESIGNATES EDGE CARD CONN.


EOC
XFER
INTADØ4
INTADØ2
INTADØ1
V1
V2
V3
V4
V5
V6
+5V
33Ω
A17 INTEGRATE CHANNEL ØØ
A19 INTEGRATE CHANNEL Ø1
A21 INTEGRATE CHANNEL Ø2
A23 INTEGRATE CHANNEL Ø3
A25 INTEGRATE CHANNEL Ø4
A27 INTEGRATE CHANNEL Ø5
A29 INTEGRATE CHANNEL Ø6
A31 INTEGRATE CHANNEL Ø7
ALL DIODES IN3064
ALL CAPACITORS 3300pf
ALL UNMARKED RESISTORS 10KΩ
V1 - SN74195N
V2 - SN7442N
V3-V6 - SN74123N

Appendix F

Program Listing

A listing of this program will be supplied on request upon payment of the necessary duplication costs.